2. Navigation

Navigation Lectures

Attached Files: robotics_for_ai_navigation_1.pdf (792.706 KB)

Assignment 2: Navigation

Please hand in your report here

Assignment

Week 2 & 3 Exercises:

1- Learning to map and localize (preparation) (1Pt):
In this exercise you will learn how to perform mapping using any robot. We use the gmapping package (slam_gmapping module) and teleoperation package (alice_teleop_keyboard) to map the
environment.
a.
- Make a launch file that runs the Gmapping and our teleoperation module
- In your report, mention the required transforms and topics for Gmapping to work properly.
- save the map in the folder ~/sudo/ros/maps/ using the map_server package

Hint: Take a look at ROS gmapping website for more information about gmapping and map_server package

b.
- Make a launch file that runs AMCL, teleoperation module, and the map that you just saved
- Set the estimated position of the robot through python code
- Move around with the teleoperation module, and report AMCL behaviour. How can you break it? Hint: Try Very fast speeds, or change base model parameters of amcl.

Hint: Take a look at the ros amcl website for a better understanding of the package, and the api to connect to it

2- Configuring and Understanding Move_base(1.5Pt):
In this exercise, you have to completely configure the navigation stack for Alice, with all the sensors. You have to describe the use of each parameter.
Hint: Try not to use pre-hydro style parameter setting.

- Main Launch file:
  - ~~The navigation stack requires the availability of a transform from base frame (e.g. base_link) to map. Make sure that required packages for this are in the launch file~~
  - The move_base launch parameters requires several parameter files (and namespaces)
    - costmap_common_parameters.yaml:
      - parameters that are accessible for all namespaces such as footprint and observation sources (sensors)
    - local_costmap_parameters.yaml:
      - parameters that are used for local_base_planner:
        - selected observation source
        - costmap size
        - publishing / vizualization params
        - costmap type
    - global_costmap_parameters.yaml:
      - parameters that are used for global_base_planner:
        - observation sources
        - map size to use
        - publish / vizualization params
        - costmap type
    - local_planner.yaml
      - Parameters for local planner:
        - Acceleration limits
        - speed limits
        - Simulation parameters, etc.

The alice_nav_sim package contains the example setup for the move_base. You can find the alice_nav.launch file in the launch folder which contains move_base, amcl, and map loading section. You also have to run the xtions.launch from the same folder, this launch file is responsible to down_sample the pointclouds from both of the 3D sensors.
You should read the ros move_base wiki page to understand how each parameter work. The move_base wiki package has separate sub-pages with information related to move_base itself,
local and global planner, costmaps, etc.
prepare the navigation stack with following properties:
~~- Command velocity of 10Hz~~
- Move_base should give up after 3 seconds if it cannot find a path
- Move_base should keep trying with local commands up to 10 seconds.
~~- Robot is not allowed to rotate on spot if it is stuck.~~

~~- Robot should clear costmap after it is stuck~~
~~- Robot should use the the navfn as the base global planner.~~
~~- Robot should use the Trajectory Rollout Planner as local planner~~
- Robot should not update costmap when it is not moving
- Robot should use a circular footprint for global planning and a rectangular footprint for
local planner
- The inflation radius for global planner should be 10cm more than the local planner.
- Robot should use all the available sensors for local planning.
- Robot may use any sensor for global planning
- The 3D sensor in local planner has higher priority than the Laser. It should overwrite the laser. Hint: use rqt_reconfigure and check move_base costmap parameters, or read the API
- The Footprint of the local planner should be convex representation of the real robot

~~In addition, explain with your own words, why there are two observation sources for the 3D sensors, one that marks, and one that clears.~~

The above issue is apparently fixed in the indigo release of move_base, so ignore it.

3- Move_base(1.5Pt):
Now that you have the navigation stack ready, it is time to use move_base available services to control the robot.
- There are 4 given locations in the locations.dat file, ~/sudo/brain/data/locations/ . Using the action-client service, order the robot to move to all of these points one after another. You have to reach all the points.
- Use the same approach as previous section, load positions from locations-dynamic.dat, and order the robot to approach them. In this scenario, there will be small obstacles on the way, and one door will be blocked. You have to be able to reach the goal anyway.
Hint: try to use the make_plan service

4- Exploring unknown areas (requires GMapping)(1Pt):
In this exercise, you will use Gmapping in combination with move_base to explore unknown environments. Replace the Localization module in your launch file with GMapping. Take the best global_planner (hint: it should be able to deal with unknown environment), and modify the local_planner parameters (hint: sticking to the global planner path is not always the best choice anymore). Use a wandering approach of your own to maximize exploration gain. Report your results

5- Understanding the local planner(1Pt):
In this exercise you will learn the effects of the local planner parameters on the robot. Use the gazebo_local_planner.launch to get the

environment for your exercise.

- Base control parameters:
  - Use the given parameter files as your base line. Use the ros graph to compare the given cmd_vel values (linear and angular speed) with the given odometry twist values (linear and angular speed).
  - Increase the acceleration limits (one by one) by 0.2 m/s^2 step by step, and report the changes, and the behaviour of the robot.
  - Decrease the acceleration limits and do the same.
  - What is the best Acceleration limit? How did you select it? Hint: You can actually measure the acceleration (a = delta V / t)
- Simulation Parameters:
  - reduce/increase the vx/vtheta sampling and report the results
  - reduce increase the simulation time and report the results
- Scoring Parameters:
  - Increase/Decrease the path/goal/obstacle scoring and report the results

6- Using the Behaviour architecture for navigating(1Pt):
- Use the navigation behaviour to reach the goals in exercise 4. After each goal is reached, run a separate behaviour that prints
something. If the reaching fails, try to approach the same goal again for 3 times, before skipping the selected goal.

7- Alice(3+1Pt):
Now that you have a good understanding of the navigation stack and its effects on the robot, you can test it on Alice.
Use the exercise 2 on Alice. Use what you have learned about the parameter files.